Decentralized Personalized Federated Min-Max Problems

Ekaterina Borodich
MIPT, Russia

Aleksandr Beznosikov
MIPT, Russia

Abdurakhmon Sadiev
MIPT, Russia

Vadim Sushko
Bosch, Germany

Alexander Gasnikov
MIPT, Russia

Abstract

Personalized Federated Learning (PFL) has recently seen tremendous progress, allowing the design of novel machine learning applications to preserve the privacy of the training data. Existing theoretical results in this field mainly focus on distributed optimization for minimization problems. This paper is the first to study PFL for saddle point problems (which cover a broader class of optimization problems), allowing for a more rich class of applications requiring more than just solving minimization problems. In this work, we consider a recently proposed PFL setting with the mixing objective function, an approach combining the learning of a global model together with locally distributed learners. Unlike most previous work, which considered only the centralized setting, we work in a more general and decentralized setup that allows us to design and analyze more practical and federated ways to connect devices to the network. We proposed new algorithms to address this problem and provide a theoretical analysis of the smooth (strongly-)convex-(strongly-)concave saddle point problems in stochastic and deterministic cases. Numerical experiments for bilinear problems and neural networks with adversarial noise demonstrate the effectiveness of the proposed methods.

1 Introduction

Distributed optimization methods have already become an integral part of solving applied problems, including many applications in machine learning. For example, distributing training data evenly across multiple devices can greatly speed up the learning process. Recently, a new research direction concerning distributed optimization - Federated Learning (FL) [23, 16], has appeared. Unlike classical distributed learning methods, the FL approach assumes that data is not stored within a centralized computing cluster but is stored on clients' devices, such as laptops, phones, tablets... This formulation of the training problem gives rise to many additional challenges, including the privacy of client's data, high heterogeneity of data stored on local devices, to name a few. In the most standard setting of distributed and federated learning, the goal is to find the parameters of a global model based on all local data.

Personalized FL. In this work, we allow each client to build their own personalized model and utilize a decentralized communication protocol that will enable harvesting information from other local models (trained on local clients' data). The problem of a prediction of the next word written on a mobile keyboard [11] has already become a typical example when the performance of a local (personalized) model is significantly ahead of the classical FL approach that trains only global model. Improving the local models using this additional knowledge may need a more careful balance, considering a possible discrepancy between data splits the local models were trained on. Attempts to find the balance between personalization and globalization have resulted in a series of works united by a common name – Personalized Federated Learning (PFL). We refer the reader to the following

1st NeurIPS Workshop on New Frontiers in Federated Learning (NFFL 2021), Virtual Meeting.

survey papers [19, 14] for more details and explanations of different techniques.

Saddle Point Problems. In this paper, we also cover the topic of PFL, but unlike previous works that focused on the minimization problem, we consider Saddle Point Problems (SPPs). SPPs cover a broader class of problems than minimization problems, and numerous important practical applications are formulated as SPPs. These include the already well-known and famous examples from game theory or optimal control [4]. In recent years, saddle point problems have become popular in several other respects. One can note a branch of recent work devoted to solving nonsmooth problems by their reformulation in the form of saddle point points [25, 24], as well as the application of such approaches in image processing [2, 3]. Most recently, significant attention of the community was devoted to saddle problems in machine learning, e.g., Generative Adversarial Networks (GANs), that are written as a min-max problem [6].

		Strongly-convex–strongly-concave		Convex–concave	
		Communications	Local computations	Communications	Local computations
Proximal	Upper	$\min\left[\frac{L}{\mu}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Alg. 1 and Alg. 2)**	$\min\left[\frac{L}{\mu};\frac{\lambda\lambda_{\max}(W)}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Alg. 1 and Alg. 2)**	$\min\left[\frac{L\Omega^2}{\varepsilon}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right]$ **(Alg. 1 and Alg. 2)**	$\min\left[\frac{L\Omega^2}{\varepsilon};\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right]$ **(Alg. 1 and Alg. 2)**
	Lower	$\min\left[\frac{L}{\mu}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Section 3)**	$\min\left[\frac{L}{\mu};\frac{\lambda\lambda_{\max}(W)}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Section 3)**	$\min\left[\frac{L\Omega^2}{\varepsilon}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right]$ **(Section 3)**	$\min\left[\frac{L\Omega^2}{\varepsilon};\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right]$ **(Section 3)**
Gradient	Upper	$\min\left[\frac{L}{\mu}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Alg. 1 and Alg. 2)**	$\frac{L}{\mu}\log\frac{1}{\varepsilon}$ **(Alg. 1 and Alg. 2)**	$\min\left[\frac{L\Omega^2}{\varepsilon}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right]$ **(Alg. 1 and Alg. 2)**	$\frac{L\Omega^2}{\varepsilon}$ **(Alg. 1 and Alg. 2)**
		$\frac{\lambda\lambda_{\max}(W)+\sqrt{L\lambda\lambda_{\max}(W)}}{\mu}\log\frac{1}{\varepsilon}$ **(Alg. 3)**	$\frac{L+\sqrt{L\lambda\lambda_{\max}(W)}}{\mu}\log\frac{1}{\varepsilon}$ **(Alg. 3)**	$\frac{\left(\lambda\lambda_{\max}(W)+\sqrt{L\lambda\lambda_{\max}(W)}\right)\Omega^2}{\varepsilon}$ **(Alg. 3)**	$\frac{\left(L+\sqrt{L\lambda\lambda_{\max}(W)}\right)\Omega^2}{\varepsilon}$ **(Alg. 3)**
	Lower	$\min\left[\frac{L}{\mu}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Section 3)**	$\frac{L}{\mu}\log\frac{1}{\varepsilon}$ **(Section 3)**	$\min\left[\frac{L\Omega^2}{\varepsilon}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right]$ **(Section 3)**	$\frac{L\Omega^2}{\varepsilon}$ **(Section 3)**
Stochastic	Upper	$\frac{\lambda\lambda_{\max}(W)}{\mu}\log\frac{1}{\varepsilon}$ **(Alg. 3)**	$r+\sqrt{r}\frac{L}{\mu}\log\frac{1}{\varepsilon}$ **(Alg. 3)**	$\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}$ **(Alg. 3)**	$\sqrt{r}\frac{L\Omega^2}{\varepsilon}$ **(Alg. 3)**
		$\frac{\lambda\lambda_{\max}(W)}{\mu}\log\frac{1}{\varepsilon}$ **(Alg. 1)**	$\left[r\frac{\lambda\lambda_{\max}(W)}{\mu}+\frac{\sqrt{r}L}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Alg. 1)**	$\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}$ **(Alg. 1)**	$r\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}+\sqrt{r}\frac{L\Omega^2}{\varepsilon}$ **(Alg. 1)**
	Lower	$\min\left[\frac{L}{\mu}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)}{\mu}\right]\log\frac{1}{\varepsilon}$ **(Section 3)**	$r+\sqrt{r}\frac{L}{\mu}\log\frac{1}{\varepsilon}$ **(Section 3)**	$\min\left[\frac{L\Omega^2}{\varepsilon}\sqrt{\chi};\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right]$ **(Section 3)**	$\sqrt{r}\frac{L\Omega^2}{\varepsilon}$ **(Section 3)**

Table 1: Summary of complexity results (upper and lower bounds) for finding an ε-solution for in the deterministic proximal (any local computations are cheap), deterministic gradient and stochastic (finite-sum) setups. In the strongly convex - strongly convex case, convergence is measured by the distance to the solution. In the convex-concave case, convergence is measured in terms of the gap function. *Notation:* μ = strong-convexity constant of f, L = constant of L-smoothness of f, Ω = diameter (in Euclidean norm) of the optimization set, $\lambda_{\max}(W)$ = maximum eigenvalue of W, $\lambda_{\min}(W)$ = minimum eigenvalue of W, $\chi = \lambda_{\max}(W)/\lambda_{\min}(W)$, r = the size of the local dataset.
green = optimal bounds (match lower bounds), blue = optimal in some cases.

1.1 Problem Formulation

Following [9, 10], we can define the Personalized Federated Saddle Point Problem (PF SPP) with a mixing objective as follows:

$$\min_{x_1,\ldots,x_M}\max_{y_1,\ldots,y_M}\left(\frac{1}{M}\sum_{m=1}^{M}f_m(x_m,y_m)+\frac{\lambda}{2M}\sum_{m=1}^{M}\|x_m-\bar{x}\|^2-\frac{\lambda}{2M}\sum_{m=1}^{M}\|y_m-\bar{y}\|^2\right). \quad (1)$$

The problem consists of the main objective function and two regularizers, where $x_1,\ldots,x_M$ and $y_1,\ldots,y_M$ are interpreted as local models on nodes, $\bar{x}=\sum_{m=1}^{M}x_m$, $\bar{y}=\sum_{m=1}^{M}y_m$ and $\lambda>0$ is the key regularization parameter, which corresponds to the degree of personalization of the models. For example, with $\lambda=0$, the problem (1) will decompose into M separable problems and each $m\in M$ will train just a local model. As λ increases, the "importance" degree of regularization terms increase and local x_m, y_m tend to $\bar{x},\bar{y}$. This formulation of the problem is valid both in the case of centralized algorithms (all nodes communicate with the main node - the server) and in the case of decentralized ones (there is no central server, all nodes are connected within a network, while only connected nodes can communicate with each other).

Decentralized setting. In the scope of the decentralized setting, it is not straightforward to calculate $\bar{x},\bar{y}$. The simplest solution to calculate them is to gather all the local values in one node, average

them, and then re-distribute them back to all the nodes. However, such an approach can still be considered a centralized case. Alternatively, to avoid the global average representation for all the nodes, it can be sufficient to find a consensus and hence find an approximate average. For such purposes gossip protocols [15, 30] can be utilized. The main difficulty of such an approach is the fact that when solving the problem (1), the gradients of the regularizer are unavailable without knowing $\bar{x}$ and $\bar{y}$. This motivated us to consider a more general formulation of the problem for the decentralized case:

$$\min_X \max_Y \{F(X,Y) := f(X,Y) + \varphi(X,Y)\} \tag{2}$$

with $f(X,Y) = \sum_{m=1}^M f_m(x_m, y_m)$, $\varphi(X,Y) = \frac{\lambda}{2}\|\sqrt{W}X\|_F^2 - \frac{\lambda}{2}\|\sqrt{W}Y\|_F^2$, where for convenience we group all local vectors x_m, y_m into matrices $X := [x_1, \ldots, x_M]^T$ and $Y := [y_1, \ldots, y_M]^T$, $\lambda > 0$, and W is the gossip matrix reflecting the properties of the communication graph between the nodes. The gossip matrix W has several interesting properties. In particular, it contains nonzero elements w_{ij} if and only if there is an edge (i,j) in the connection graph. Unlike (1), the formulation (2) penalizes not the difference with the global average, but the difference with other connected local nodes. Moreover, this proposed formulation enables computing the gradients of $\varphi : \nabla_X \varphi(X,Y) = \lambda W X$ and $\nabla_Y \varphi(X,Y) = -\lambda W Y$, for which just one communication with neighbors is enough. The idea of using this type of penalty is not new and has been used in the literature in several contexts, in particular for classical decentralized minimization [21, 7] with large λ and for multitask PFL [28, 29] with small λ. In this paper, we focus on the case when λ is a small parameter. Note that in the proposed formulation (2) we consider both the centralized[1] and decentralized cases.

Notation: For vectors, we use the Euclidean norm $\|\cdot\| = \|\cdot\|_2$ everywhere, and for matrices - the Frobenius norm $\|\cdot\|_F$. We define a projection operator $\text{proj}_{\mathcal{C}}(x) = \min_{u \in \mathcal{C}} \|u - x\|$ which is the Euclidean projection onto $\mathcal{C}$. We denote by I the identity matrix and by E an all-ones matrix. For the gossip matrix W we denote $\lambda_{\max}(W) = \lambda_1(W) \geq \ldots \geq \lambda_M(W) = 0$ the spectrum of W.

For given convex-concave function $g(x,y)$, convex sets X, Y and any $x \in X, y \in Y$ we define proximal operator as follows: $\text{prox}_g(x,y) = \arg\min_{u \in X} \max_{v \in Y} \{g(u,v) + \frac{1}{2}\|x-u\|^2 - \frac{1}{2}\|y-v\|^2\}$. Let us define the functions of these matrices, as it was done in (2). We introduce the following notation for gradients:

$$\nabla_X f(X,Y) = [\nabla_{x_1} f_1(x_1, y_1), \ldots, \nabla_{x_M} f_M(x_M, y_M)]^T,$$
$$\nabla_Y f(X,Y) = [\nabla_{y_1} f_1(x_1, y_1), \ldots, \nabla_{y_M} f_M(x_M, y_M)]^T,$$
$$\nabla_X \varphi(X,Y) = \lambda W X, \quad \nabla_Y \varphi(X,Y) = -\lambda W Y.$$

1.2 Summary of Contributions

This paper is the first paper (to the best of our knowledge) that proposes optimal algorithms and derives the computational and communication lower bounds for SPPs in a PFL setting. We now outline the *main* contribution of our work as follows:

• We present a new SPP formulation of the PFL problem (2) as the decentralized min-max mixing model. This extends the classical PFL problem to a broader class of problems beyond the classical minimization problem. It furthermore covers various comm. topologies and hence goes beyond the centralized setting.

• We develop a lower bound both for the amount of communication and local computation for a general class of algorithms (that satisfy Assumption 3). The bounds naturally depend on the communication matrix W (as in the minimization problem), but our results apply to SPP (see "Lower" rows in Table 1 for various settings of the SPP PFL formulations).

• We develop multiple novel algorithms to solve SPP for optimizing min-max decentralized personalized federated learning problems (2). The first methods (Algorithm 1 and 2) is based on recent sliding method [20, 27, 9], but is extended to hangle SPPs in a decentralized PFL. The second method (Algorithm 3) extends recently proposed randomized local method of [10] for optimizing SPP in PFL setting.

• We provide the theoretical convergence analysis for all proposed algorithms. Our theoretical results provide the oracle complexities concerning both the function f and the function φ. For example, some settings require that we perform $\tilde{\mathcal{O}}\left(\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right)$ communication rounds (calls of $\nabla_X \varphi(X,Y)$,

[1] The centralized case corresponds to a complete computational graph. If we set W to the Laplacian of a complete graph, it is easy to verify that we obtain (1).

$\nabla_Y \varphi(X,Y)$) and $\tilde{\mathcal{O}}\left(\frac{L\Omega^2}{\varepsilon}\right)$ local computations on each node (calls of $\nabla_X f(X,Y)$, $\nabla_Y f(X,Y)$) (see Table 1 for overview of developed complexity results).

• We adapt the proposed algorithm for training neural networks. We provide an explanation and intuition of how our algorithm could be adapted to the neural network settings. We compare our algorithms: type of sliding and type of local method. To the best of our knowledge, this is the first work that compares these approaches in the scope of neural networks, as previous studies were limited to simpler methods, such as regression problems [10, 9]. Our experiments confirm the robustness of our methods on the problem of training a classifier with adversarial noise.

2 Assumptions

Before presenting the lower bound complexity results, we state key assumptions about the problem statement as well as the connection graph.

Functions and domains. We consider problem (2) on convex compact domains $\mathcal{X}$ and $\mathcal{Y}$, i.e. all $x_m \in \mathcal{X} \subset \mathbb{R}^{n_x}$, all $y_m \in \mathcal{Y} \subset \mathbb{R}^{n_y}$. Let the set $\mathcal{X} \times \mathcal{Y}$ has diameter Ω. Additionally, we introduce standard assumptions on f_m and f.

Assumption 1

• *Each f_m is L-smooth on $\mathcal{X} \times \mathcal{Y}$, i.e. for all $x_1, x_2 \in \mathcal{X}$, $y_1, y_2 \in \mathcal{Y}$ it holds that* $\|\nabla_x f_m(x_1,y_1) - \nabla_x f_m(x_2,y_2)\|^2 + \|\nabla_y f_m(x_1,y_1) - \nabla_y f_m(x_2,y_2)\|^2 \leq L^2\left(\|x_1-x_2\|^2 + \|y_1-y_2\|^2\right)$; *in case, when* $f_m = \frac{1}{r}\sum_{i=1}^r f_{m,i}$, f_m *is L-average smooth, i.e.* $\frac{1}{r}\sum_{i=1}^r \|\nabla_x f_{m,i}(x_1,y_1) - \nabla_x f_{mi}(x_2,y_2)\|^2 + \|\nabla_y f_{m,i}(x_1,y_1) - \nabla_y f_{mi}(x_2,y_2)\|^2 \leq L^2\left(\|x_1-x_2\|^2 + \|y_1-y_2\|^2\right)$;

• *f is μ - strongly-convex-strongly-concave on $\mathcal{X} \times \mathcal{Y}$, i.e. for all $x_1, x_2 \in \mathcal{X}$, $y_1, y_2 \in \mathcal{Y}$ it holds that* $\langle \nabla_x f(x_1,y_1) - \nabla_x f(x_2,y_2); x_1 - x_2\rangle - \langle \nabla_y f(x_1,y_1) - \nabla_y f(x_2,y_2); y_1 - y_2\rangle \geq 2\mu\left(\|x_1-x_2\|^2 + \|y_1-y_2\|^2\right)$;

• *Each f_m is convex-concave on $\mathcal{X} \times \mathcal{Y}$, i.e. 0- strongly-convex-strongly-concave.*

Communication network. The communication network between devices can be represented as a fixed, connected, undirected graph. $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, where the set of vertices $\mathcal{V}$ represents the set of computing nodes, and the edges $\mathcal{E}$ indicate the presence or absence of connection between the corresponding nodes. Recall that we consider a decentralized case, where information exchange (by gossip protocol) is possible only between neighbors. In such a case, communication is represented as a matrix multiplication with a matrix W introduced in (2). It remains to introduce a formal definition of gossip matrix:

Assumption 2 *We call a matrix W a gossip matrix if it satisfies the following conditions: 1) W is an $M \times M$ symmetric, 2) W is positive semi-definite, 3) ker$(W) =$ span$\{(1,\ldots,1)\}$, 4) W is defined on the edges of the network: $w_{ij} \neq 0$ only if $i = j$ or $(i,j) \in \mathcal{E}$.*

Computing oracles. For local computations we introduce three different oracles: proximal, gradient and gradient of summands. More formally, for each device i and local points x_m, y_m we can compute one of the following

• **proximal oracle** ($\gamma_m \geq 0$): $\text{Loc}(x_i,y_i,i) = \left\{ \left(\nabla_X f_i(x_i,y_i), -\nabla_Y f_i(x_i,y_i)\right)^T, \text{prox}_{\gamma_i f_i}(x_i,y_i)\right\}$

• **gradient oracle**: $\text{Loc}(x_i,y_i,i) = \left\{\left(\nabla_X f_i(x_i,y_i), -\nabla_Y f_i(x_i,y_i)\right)^T\right\}$

• **summand gradient oracle**: $\text{Loc}(x_i,y_i,i) = \left\{\left(\nabla_X f_{i,j_i}(x_i,y_i), -\nabla_Y f_{ij_i}(x_i,y_i)\right)^T\right\}$.

In fact, by proximal oracle, we can quickly and cheaply solve any local subproblems (compute prox). Gradient oracle gives access to local gradients. Summand gradient refers to the stochastic case, when of local functions have finite-sum structure: $f_i = \frac{1}{r}\sum_{j=1}^r f_{i,j}$, and in each call of oracle we can compute gradients of only summand (selected randomly or deterministically).

3 Lower Bounds

Before presenting the lower complexity bound (for both communication and local computation) for solving the problem (2), we first define a class of algorithms for which the lower bound will be proved. Note that the communication oracle allows devices to collect information from others with the gossip weight matrix W. The local computation oracle is defined above. The class of algorithms analyzed will satisfy the following assumption:

Assumption 3 *Let $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k=1}^{\infty}$ be iterates generated by algorithm $\mathcal{A}$. For each node of graph G we define sequence of local memory $\{\mathcal{M}_{m,k}\}_{k=1}^{\infty}$ for $1 \leq m \leq M$: $\mathcal{M}_{m,0} = span\left\{(x_m^0, y_m^0\}\right)$ and*

$$\mathcal{M}_{m,k+1} = \begin{cases} span\left\{\mathcal{M}_{m,k}, \{(x_m^k, y_m^k), Loc(\mathbf{x}^k, \mathbf{y}^k, m)\}\right\}, & \text{if local oracle at the iteration } k \\ span\left\{\bigcup_{j:(m,j)\in\mathcal{E}} \mathcal{M}_{j,k}\right\}, & \text{if consensus oracle.} \end{cases}$$

3.1 Lower complexity bounds on the communications

The following theorem presents the lower bound for the communication complexity:

Theorem 1 *Let $\sqrt{\chi} \geq 6$, $L \geq \mu$, $\lambda\lambda_{\min}^{+}(W) \geq \mu$. Then, there exists a graph G, with a corresponding matrix W satisfying Assumption 2, functions $f_1, f_2, \ldots, f_M : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ satisfying Assumption 1, and a starting point (x^0, y^0) such that for the sequences of iterates $\{(x_m^k, y_m^k)\}_{k=1}^{N}$ generated by any algorithm $\mathcal{A}$, satisfying Assumption 3, for any device m it holds*

$$\|x_m^N - x^*\|^2 + \|y_m^N - y^*\|^2 \geq \left(1 - 10\max\left\{\frac{\mu}{\lambda\lambda_{\max}(W)}, \frac{\mu}{(L-\mu)\sqrt{\chi}}\right\}\right)^S \frac{\|y^0 - y^*\|^2}{4},$$

where N is number of oracle calls, $S \leq N$ is number of consensus oracle calls.

3.2 Lower complexity bounds on the local oracle calls

Next, we present the lower complexity bounds on the number of the local oracle calls for different types of a local oracle.

Proximal oracle. The construction we did in Theorem 1 requires $O\left(\left\{\frac{\min\{\lambda\lambda_{\max}(W), L\sqrt{\chi(W)}\}}{\mu}\log\frac{1}{\varepsilon}\right\}\right)$ communication rounds to reach ε-solution of (2), but on top of that, it also requires at least $O\left(\left\{\frac{\min\{\lambda\lambda_{\max}(W), L\}}{\mu}\log\frac{1}{\varepsilon}\right\}\right)$ calls of local oracle. Accessing the local gradients is naturally needed for the algorithm to move closer to the optimal solution. Hence, this concludes the lower bound on the case of using local proximal oracle.

Gradient oracle. Starting with $(X_0, Y_0) = (0, 0)$, in case when matrix W is a Laplace matrix of a fully connected graph and choosing $f_1 = f_2 = \cdots = f_M$, the problem (2) reduces to min-max a single local function f_1. From [31], we know that the worst-case need at least $O\left(\frac{L}{\mu}\log\frac{1}{\varepsilon}\right)$ gradient calls to find ε-solution [31]. Hence, we can obtain the lower bound for gradient oracle. Because we start from the same starting point on each node, and all the local functions are identical, the communication does not help in the convergence. The construction of f only allows exploring a single coordinate per a local call, regardless of the comm.

Summand gradient oracle. Let us assume that algorithm $\mathcal{A}$ is either deterministic, or generated by given seed that is initialized identically for all clients. The same way as for gradient oracle let us set $(X_0, Y_0) = (0, 0)$, W is the Laplace matrix of a fully connected graph and $f_1 = f_2 = \cdots = f_M$. For this algorithm, all local iterates will be identical, i.e., $(x_1^k, y_1^k) = (x_2^k, y_2^k) = \cdots = (x_M^k, y_M^k)$ for all $k \geq 0$. Consequently, the problem reduces to min-max of a single finite sum objective f_1, which needs at least $O((r + \frac{\sqrt{r}L}{\mu})\log\frac{1}{\varepsilon})$ summand gradient calls [8] to get solution with error ε.

4 Optimal algorithms

In this section, we present optimal algorithms for solving the problem (2). In the main paper, we include only a part of the contribution. We consider only one type of methods (Sliding), only in the case when $\lambda\lambda_{\max}(W) \leq L\sqrt{\chi(W)}$ and with one type of oracles (deterministic). In Appendix A, you can find the rest of the contribution. See Table 1 for summary.

4.1 PF Min-Max via Sliding

4.1.1 Case $\lambda\lambda_{\max}(W) \leq L\sqrt{\chi(W)}$

The simplest method for solving (2) is to consider the function F as a whole, not to take into account its composite structure. As a basic method, for example, the classical method for smooth saddle point

Algorithm 1 Sliding 1 for Decentralized Min-Max(S1DMM)

parameters: stepsize γ, precision δ
initialization: choose $x^0, y^0 \in \mathcal{X} \times \mathcal{Y}$, $x_m^0 = x^0$, $y_m^0 = y^0$ for all m

1: **for** $k = 0, 1, 2, \ldots$ **do**
2: $V_x^k = X^k - \gamma \cdot \lambda W X^k, \quad V_y^k = Y^k - \gamma \cdot \lambda W Y^k$
3: Find U^k, such that $\|U^k - \hat{U}^k\|_F^2 \leq \delta$, where $\hat{U}^k$ is a solution of:

$$\min_{U_x} \max_{U_y} \gamma f(U_x, U_y) + \frac{\|U_x - V_x^k\|_F^2}{2} - \frac{\|U_y - V_y^k\|_F^2}{2} \tag{3}$$

4: $X^{k+1} = \text{proj}_{\mathcal{X}} \left(U_x^k + \gamma \cdot \lambda \left(W X^k - W U_x^k \right) \right)$
5: $Y^{k+1} = \text{proj}_{\mathcal{Y}} \left(U_y^k + \gamma \cdot \lambda \left(W Y^k - W U_y^k \right) \right)$
6: **end for**

problems - Extra Step Method [17] (or Mirror Prox [13]) - is suitable. Then the number of calls to the oracle for f andφ are the same. In our case (see the explanation above) the step along the gradient of the regularizer requires communication with neighboring nodes. To calculate the gradients of f, it is enough to calculate all the local gradients of f_m and do not exchange information at all. Of course, we want to reduce the number of communications (and calls the regularizer gradient) as much as possible. This is especially true when we want a fairly personalized model ($\lambda \ll L$) and information from other nodes is not particularly important. To solve this problem and separate the oracle complexities for function f and function φ, we base our method on *Sliding technique* [20]. The optimal method for PF minimization from [9] is also a kind of Sliding method.
It is clear that for saddle point problems, we cannot just use the method from [9]. Sliding for saddles has its own specifics [27] – exactly for the same reasons why Extra Step Method is used for smooth saddles instead of the usual Descent-Ascent [5] (at least because Descent-Ascent diverges for the most common bilinear problems). Let us give some comments on how the Algorithm 1 works. Note that we only need to communicate with other devices on lines 2, 4 and 5: devices send variables x_m^k and compute WX^k and WU_x^k. Each device m does not compute the full product of matrices, it calculates only the corresponding line: $v_{x,m}^k = x_m^k - \gamma \cdot \lambda \sum_{i=1}^M w_{m,i} x_i^k$. This step requires only information from the node's neighbors (similarly written for V_y^k and U^k). Furthermore, we note that subproblem (3) is solved locally and separately on each machine (in parallel). Indeed, one can divide it on M disjoint problems: $\min_{u_{x,m}} \max_{u_{y,m}} \gamma f_m(u_{x,m}, u_{y,m}) + \frac{1}{2}\|u_{x,m} - v_{x,m}^k\|^2 - \frac{1}{2}\|u_{y,m} - v_{y,m}^k\|^2$. Each of these problems can be resolved locally, for example, by Extra Step Method [13] or by Randomized Extra Step Method [1]. In lines 4 and 5, under the projection of matrices, we mean that each row of the matrix is projected onto the set.
• **S1DMM + Extra Step Method.** The following theorem states the convergence rate of S1DMM with Extra Step Method [13] as a local algorithm for subproblem (3).

Theorem 2 *Let Algorithm 1 be applied for solving (2) with convex-concave and L -smooth local functions f_m. Then one can choose a constant step γ, a precision δ, so that we need $\mathcal{O}\left(\frac{\lambda\lambda_{\max}(W)\Omega^2}{\varepsilon}\right)$ communication rounds and $\tilde{\mathcal{O}}\left(\frac{L\Omega^2}{\varepsilon}\right)$ local computation on each node, to obtain $\hat{z} = (\hat{x}, \hat{y})$ such that $[\max_{y \in \mathcal{Y}} f(\hat{x}, y) - \min_{x \in \mathcal{X}} f(x, \hat{y})] \leq \varepsilon$.*
If we additionally assume that global objective function f is μ-strongly-convex-strongly-concave, then after $\mathcal{O}(\frac{\lambda\lambda_{\max}(W)}{\mu} \log \frac{1}{\varepsilon})$ communication rounds and $\tilde{\mathcal{O}}(\frac{L}{\mu} \log \frac{1}{\varepsilon})$ local computations on each node we will obtain $\hat{z}$, such that $\|\hat{z} - z^*\|^2 \leq \varepsilon$.

As expected, the communication complexity of S1DMM + Extra Step Method is $O\left(\frac{\lambda\lambda_{\max}(W)}{\mu} \log \frac{1}{\varepsilon}\right)$ in the strongly-convex-strongly-concave case, thus optimal when $L\sqrt{\chi(W)} = O(\lambda\lambda_{\max}(W))$. The local gradient complexity is $\tilde{O}(\frac{L}{\mu})$, which is, up to log and constant factors identical to the lower bound on the local gradient calls. To get an estimate for the proximal oracle, it is enough to note that the subproblem (3) is local and matches the definition of the proximal oracle, then we can solve this problem in one oracle call.

5 Experiments

The goal of this experiment is to compare our new methods: Algorithm 1 and Algorithm 3 for a neural network complemented with a robust loss [22]

$$f_m(x_m, y_m) := \frac{1}{N_m}\sum_{n=1}^{N_m} \ell(g(x_m, a_n + y_n), b_n) + \frac{\beta_x}{2}\|x_m\|^2 - \frac{\beta_y}{2}\|y_m\|^2,$$

where x_m are the weights of the mth model, $\{(a_n, b_n)\}_{n=1}^{N_m}$ are pairs of the training data on the mth node, y_n is the so-called adversarial noise, which introduces a small effect of perturbation in the data. In Appendix B.3 we discuss connection between neural networks and our Algorithms.

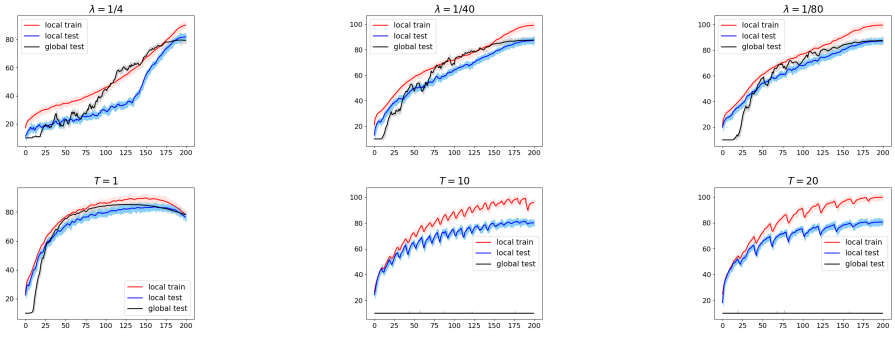
Data and model. We consider the benchmark of image classification on the CIFAR-10 [18] dataset. It contains 50000 and 10000 images in the training and validation sets respectively, equally distributed over 10 classes. To emulate the distributed scenario, we partition the dataset into N non-overlapping subsets in a heterogeneous manner. For each subset, we select a major class that forms 25% of the data, while the rest of the data split is filled uniformly by other classes. We parameterize each of the learners as a convolutional neural network, taking ResNet-18 [12] architecture as the backbone. As a loss function, we use multi-class cross-entropy, complemented with an adversarial noise.

Setting. To train ResNet18 in CIFAR-10, one can use stochastic gradient descent with momentum 0.9, the learning rate of 0.1 and a batch size of 128. We will use our two Algorithms in intuition from Appendix B.3. In order for the comparison of Algorithm 1 and Algorithm 3 to be fair, it is necessary to equalize the number of communications and local iterations for both algorithms, that is why we need carefully choose T (the number of inner/local iterations in Algorithm 1) and p (probability in Algorithm 3). Tables 2 and 3 shows all the experimental setups that we consider.

λ	T	θ	q	Results
$1/4$	40 (1 epoch)	2	2	Figure 1 (a)
$1/40$	400 (10 epochs)	2	2	Figure 1 (b)
$1/80$	800 (20 epochs)	2	2	Figure 1 (c)

Table 2: Additional parameters for comparison of Algorithm 1 and Algorithm 3.

Results. One can see the results of the experiment in Figure 1. Table 2 shows the correspondence of the various settings to the parameters.



(a) (b) (c)

Figure 1: Average accuracy in during process of learning with different average parameters λ and T. The first line presents the results of Algorithm 1, the second - Algorithm 3. Red line – accuracy of the local model on local train data, blue line - accuracy of the local model on local test data, black line – accuracy of the global model on global test data. The experiment was repeated 5 times, the deviations are reflected.

Discussions. We compare Algorithms based on the balance of the local and global models, i.e. if the Algorithm is able to train well both local and global models, then we find the FL balance by this Algorithm. The results show that the Local SGD technique (Algorithm 3) outperformed the Algorithm 1 only with a fairly frequent device communication (Figure 1 (a)). In other cases (Figure 1 (b), (c)), Algorithm 3 was unable to train the global model, although it withstood the good quality of the local models. It turns out that the technique of Algorithm 1 (more precisely, its intuition from Appendix B.3) can be considered robust for Federated Learning, even in the case of neural networks.

References

[1] Ahmet Alacaoglu and Yura Malitsky. Stochastic variance reduction for variational inequality methods. *arXiv preprint arXiv:2102.08352*, 2021.

[2] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40(1):120–145, 2011.

[3] Ernie Esser, Xiaoqun Zhang, and Tony F Chan. A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science. *SIAM Journal on Imaging Sciences*, 3(4):1015–1046, 2010.

[4] F. Facchinei and J.S. Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2007.

[5] Gauthier Gidel, Hugo Berard, Gaëtan Vignoud, Pascal Vincent, and Simon Lacoste-Julien. A variational inequality perspective on generative adversarial networks. *arXiv preprint arXiv:1802.10551*, 2018.

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[7] Eduard Gorbunov, Darina Dvinskikh, and Alexander Gasnikov. Optimal decentralized distributed algorithms for stochastic convex optimization. *arXiv preprint arXiv:1911.07363*, 2019.

[8] Yuze Han, Guangzeng Xie, and Zhihua Zhang. Lower complexity bounds of finite-sum optimization problems: The results and construction. *arXiv preprint arXiv:2103.08280*, 2021.

[9] Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. *arXiv preprint arXiv:2010.02372*, 2020.

[10] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020.

[11] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[13] Anatoli Juditsky, Arkadi Nemirovski, and Claire Tauvel. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems*, 1(1):17–58, 2011.

[14] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[15] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491. IEEE, 2003.

[16] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[17] G. M. Korpelevich. The extragradient method for finding saddle points and other problems. 1976.

[18] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[19] Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of personalization techniques for federated learning. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 794–797. IEEE, 2020.

[20] Guanghui Lan. Gradient sliding for composite optimization. *Mathematical Programming*, 159(1):201–235, 2016.

[21] Huan Li, Cong Fang, Wotao Yin, and Zhouchen Lin. Decentralized accelerated gradient methods with increasing penalty parameters. *IEEE Transactions on Signal Processing*, 68:4855–4870, 2020.

[22] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[24] Arkadi Nemirovski. Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004.

[25] Yu Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. 2019.

[27] Alexander Rogozin, Pavel Dvurechensky, Darina Dvinkikh, Alexander Beznosikov, Dmitry Kovalev, and Alexander Gasnikov. Decentralized distributed optimization for saddle point problems. *arXiv preprint arXiv:2102.07758*, 2021.

[28] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning. *arXiv preprint arXiv:1705.10467*, 2017.

[29] Weiran Wang, Jialei Wang, Mladen Kolar, and Nathan Srebro. Distributed stochastic multi-task learning with graph regularization. *arXiv preprint arXiv:1802.03830*, 2018.

[30] Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems and Control Letters*, 53(1):65–78, 2004.

[31] Junyu Zhang, Mingyi Hong, and Shuzhong Zhang. On lower iteration complexity bounds for the saddle point problems. *arXiv preprint arXiv:1912.07481*, 2019.